Package ‘singcar’

July 23, 2025

Title Comparing Single Cases to Small Samples

Version 0.1.5

Description When comparing single cases to control populations and no parameters are known researchers and clinicians must estimate these with a control sample. This is often done when testing a case's abnormality on some variable or testing abnormality of the discrepancy between two variables. Appropriate frequentist and Bayesian methods for doing this are here implemented, including tests allowing for the inclusion of covariates. These have been developed first and foremost by John Crawford and Paul Garthwaite, e.g. in Crawford and Howell (1998) <doi:10.1076/clin.12.4.482.7241>, Crawford and Garthwaite (2005) <doi:10.1037/0894-4105.19.3.318>, Crawford and Garthwaite (2007) <doi:10.1080/02643290701290146> and Crawford, Garthwaite and Ryan (2011) <doi:10.1016/j.cortex.2011.02.017>. The package is also equipped with power calculators for each method.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Depends R (>= 2.10)

Imports stats, CholWishart, MASS, withr

Suggests knitr, rmarkdown, bookdown, lme4, lmerTest, testthat (>= 2.1.0)

VignetteBuilder knitr

URL `https://github.com/jorittmo/singcar`

BugReports `https://github.com/jorittmo/singcar/issues`

NeedsCompilation no

Author Jonathan Rittmo [aut, cre] (ORCID: <`https://orcid.org/0000-0001-5075-0166`>),
Robert McIntosh [aut] (ORCID: <`https://orcid.org/0000-0002-7615-6699`>)

Maintainer Jonathan Rittmo <`j.rittmo@gmail.com`>

Repository CRAN

Date/Publication 2023-03-16 09:10:02 UTC

Contents

`BSDT` *Bayesian Standardised Difference Test*

Description

A test on the discrepancy between two tasks in a single case, by comparison to the discrepancy of means in the same two tasks in a control sample. Can take both tasks measured on the same scale with the same underlying distribution or tasks measured on different scales by setting `unstandardised` to `TRUE` or `FALSE` (default). Calculates a standardised effects size of task discrepancy as well as a point estimate of the proportion of the control population that would be expected to show a more extreme discrepancy as well as relevant credible intervals. This test is based on random number generation which means that results may vary between runs. This is by design and the reason for not using `set.seed()` to reproduce results inside the function is to emphasise the randomness of the test. To get more accurate and stable results please increase the number of iterations by increasing `iter` whenever feasible. Developed by Crawford and Garthwaite (2007).

Usage

```
BSDT(
  case_a,
  case_b,
  controls_a,
  controls_b,
  sd_a = NULL,
  sd_b = NULL,
  sample_size = NULL,
```

```
  r_ab = NULL,
  alternative = c("two.sided", "greater", "less"),
  int_level = 0.95,
  iter = 10000,
  unstandardised = FALSE,
  calibrated = TRUE,
  na.rm = FALSE
)
```

Arguments

`case_a`	Case's score on task A.
`case_b`	Case's score on task B.
`controls_a`	Controls' scores on task A. Takes either a vector of observations or a single value interpreted as mean. *Note*: you can supply a vector as input for task A while mean and SD for task B.
`controls_b`	Controls' scores on task A. Takes either a vector of observations or a single value interpreted as mean. *Note*: you can supply a vector as input for task B while mean and SD for task A.
`sd_a`	If single value for task A is given as input you must supply the standard deviation of the sample.
`sd_b`	If single value for task B is given as input you must supply the standard deviation of the sample.
`sample_size`	If A or B is given as mean and SD you must supply the sample size. If controls_a is given as vector and controls_b as mean and SD, sample_size must equal the number of observations in controls_a.
`r_ab`	If A or B is given as mean and SD you must supply the correlation between the tasks.
`alternative`	A character string specifying the alternative hypothesis, must be one of `"two.sided"` (default), `"greater"` or `"less"`. You can specify just the initial letter. Since the direction of the expected effect depends on which task is set as A and which is set as B, be very careful if changing this parameter.
`int_level`	Level of confidence for credible intervals, defaults to 95%.
`iter`	Number of iterations, defaults to 10000. Greater number gives better estimation but takes longer to calculate.
`unstandardised`	Estimate z-value based on standardised or unstandardised task scores. Set to `TRUE` only if tasks are measured on the same scale with the same underlying distribution.
`calibrated`	`TRUE` is default. Whether or not to use the standard theory (Jeffreys) prior distribution (if set to `FALSE`) or a calibrated prior examined by Berger and Sun (2008). The sample estimation of the covariance matrix is based on the sample size being n - 1 when the calibrated prior is used. See Crawford et al. (2011) for further information. Calibrated prior is recommended.
`na.rm`	Remove `NA`s from controls.

Details

Uses random generation of inverse wishart distributions from the CholWishart package (Geoffrey Thompson, 2019).

Value

A list with class `"htest"` containing the following components:

`statistic`	the mean z-value over `iter` number of iterations.
`parameter`	the degrees of freedom used to specify the posterior distribution.
`p.value`	the mean p-value over `iter` number of iterations.
`estimate`	case scores expressed as z-scores on task A and B. Standardised effect size (Z-DCC) of task difference betwee
`null.value`	the value of the difference under the null hypothesis.
`alternative`	a character string describing the alternative hypothesis.
`method`	a character string indicating what type of test was performed.
`data.name`	a character string giving the name(s) of the data

References

Berger, J. O., & Sun, D. (2008). Objective Priors for the Bivariate Normal Model. *The Annals of Statistics, 36*(2), 963-982. JSTOR.

Crawford, J. R., & Garthwaite, P. H. (2007). Comparison of a single case to a control or normative sample in neuropsychology: Development of a Bayesian approach. *Cognitive Neuropsychology, 24*(4), 343-372. doi:10.1080/02643290701290146

Crawford, J. R., Garthwaite, P. H., & Ryan, K. (2011). Comparing a single case to a control sample: Testing for neuropsychological deficits and dissociations in the presence of covariates. *Cortex, 47*(10), 1166-1178. doi:10.1016/j.cortex.2011.02.017

Geoffrey Thompson (2019). CholWishart: Cholesky Decomposition of the Wishart Distribution. R package version 1.1.0. `https://CRAN.R-project.org/package=CholWishart`

Examples

```
BSDT(-3.857, -1.875, controls_a = 0, controls_b = 0, sd_a = 1,
sd_b = 1, sample_size = 20, r_ab = 0.68, iter = 100)

BSDT(case_a = size_weight_illusion[1, "V_SWI"], case_b = size_weight_illusion[1, "K_SWI"],
 controls_a = size_weight_illusion[-1, "V_SWI"],
 controls_b = size_weight_illusion[-1, "K_SWI"], iter = 100)
```

`BSDT_cov` *Bayesian Standardised Difference Test with Covariates*

Description

Takes two single observations from a case on two variables (A and B) and compares their standardised discrepancy to the discrepancies of the variables in a control sample, while controlling for the effects of covariates, using Bayesian methodology. This test is used when assessing a case conditioned on some other variable, for example, assessing abnormality of discrepancy when controlling for years of education or sex. Under the null hypothesis the case is an observation from the distribution of discrepancies between the tasks of interest coming from observations having the same score as the case on the covariate(s). Returns a significance test, point and interval estimates of difference between the case and the mean of the controls as well as point and interval estimates of abnormality, i.e. an estimation of the proportion of controls that would exhibit a more extreme conditioned score. This test is based on random number generation which means that results may vary between runs. This is by design and the reason for not using `set.seed()` to reproduce results inside the function is to emphasise the randomness of the test. To get more accurate and stable results please increase the number of iterations by increasing `iter` whenever feasible. Developed by Crawford, Garthwaite and Ryan (2011).

Usage

```
BSDT_cov(
  case_tasks,
  case_covar,
  control_tasks,
  control_covar,
  alternative = c("two.sided", "greater", "less"),
  int_level = 0.95,
  calibrated = TRUE,
  iter = 10000,
  use_sumstats = FALSE,
  cor_mat = NULL,
  sample_size = NULL
)
```

Arguments

`case_tasks`	A vector of length 2. The case scores from the two tasks.
`case_covar`	A vector containing the case scores on all covariates included.
`control_tasks`	A matrix or dataframe with 2 columns and n rows containing the control scores for the two tasks. Or if `use_sumstats` is set to `TRUE` a 2x2 matrix or dataframe containing summary statistics where the first column represents the means for each task and the second column represents the standard deviation.
`control_covar`	A matrix or dataframe containing the control scores on the covariates included. Or if `use_sumstats` is set to `TRUE` a matrix or dataframe containing summary

	statistics where the first column represents the means for each covariate and the second column represents the standard deviation.
`alternative`	A character string specifying the alternative hypothesis, must be one of `"two.sided"` (default), `"greater"` or `"less"`. You can specify just the initial letter. Since the direction of the expected effect depends on which task is set as A and which is set as B, be very careful if changing this parameter.
`int_level`	The probability level on the Bayesian credible intervals, defaults to 95%.
`calibrated`	Whether or not to use the standard theory (Jeffreys) prior distribution (if set to `FALSE`) or a calibrated prior examined by Berger and Sun (2008). The sample estimation of the covariance matrix is based on the sample size being n - 1 when the calibrated prior is used. See Crawford et al. (2011) for further information. Calibrated prior is recommended.
`iter`	Number of iterations to be performed. Greater number gives better estimation but takes longer to calculate. Defaults to 10000.
`use_sumstats`	If set to `TRUE`, `control_tasks` and `control_covar` are treated as matrices with summary statistics. Where the first column represents the means for each variable and the second column represents the standard deviation.
`cor_mat`	A correlation matrix of all variables included. NOTE: the two first variables should be the tasks of interest. Only needed if `use_sumstats` is set to `TRUE`.
`sample_size`	An integer specifying the sample size of the controls. Only needed if `use_sumstats` is set to `TRUE`.

Details

Uses random generation of inverse wishart distributions from the CholWishart package (Geoffrey Thompson, 2019).

Value

A list with class `"htest"` containing the following components:

`statistic`	the average z-value over `iter` number of iterations.
`parameter`	the degrees of freedom used to specify the posterior distribution.
`p.value`	the average p-value over `iter` number of iterations.
`estimate`	case scores expressed as z-scores on task A and B. Standardised effect size (Z-DCCC) of task difference betw
`null.value`	the value of the difference between tasks under the null hypothesis.
`interval`	named numerical vector containing level of confidence and confidence intervals for both effect size and p-valu
`desc`	data frame containing means and standard deviations for controls as well as case scores.
`cor.mat`	matrix giving the correlations between the tasks of interest and the covariates included.

`sample.size`	number of controls.
`alternative`	a character string describing the alternative hypothesis.
`method`	a character string indicating what type of test was performed.
`data.name`	a character string giving the name(s) of the data

References

Berger, J. O., & Sun, D. (2008). Objective Priors for the Bivariate Normal Model. *The Annals of Statistics, 36*(2), 963-982. JSTOR.

Crawford, J. R., Garthwaite, P. H., & Ryan, K. (2011). Comparing a single case to a control sample: Testing for neuropsychological deficits and dissociations in the presence of covariates. *Cortex, 47*(10), 1166-1178. doi:10.1016/j.cortex.2011.02.017

#' Geoffrey Thompson (2019). CholWishart: Cholesky Decomposition of the Wishart Distribution. R package version 1.1.0. `https://CRAN.R-project.org/package=CholWishart`

Examples

```
BSDT_cov(case_tasks = c(size_weight_illusion[1, "V_SWI"],
                        size_weight_illusion[1, "K_SWI"]),
         case_covar = size_weight_illusion[1, "YRS"],
         control_tasks = cbind(size_weight_illusion[-1, "V_SWI"],
                               size_weight_illusion[-1, "K_SWI"]),
         control_covar = size_weight_illusion[-1, "YRS"], iter = 100)
```

`BSDT_cov_power` *Power calculator for BSDT_cov*

Description

Computationally intense. Lower `iter` and/or `nsim` for faster but less precise calculations. Calculates approximate power, given sample size, using Monte Carlo simulation for BSDT with covariates for specified (expected) case score, means and standard deviations for the control sample on the task of interest and included covariates. The number of covariates defaults to 1, means and standard deviations for the tasks and covariate default to 0 and 1, so if no other values are given the case scores is interpreted as deviation from the mean in standard deviations for both tasks and covariates.

Usage

```
BSDT_cov_power(
  case_tasks,
  case_cov,
  control_tasks = matrix(c(0, 0, 1, 1), ncol = 2),
```

```
  control_covar = c(0, 1),
  cor_mat = diag(3) + 0.3 - diag(c(0.3, 0.3, 0.3)),
  sample_size,
  alternative = c("two.sided", "greater", "less"),
  alpha = 0.05,
  nsim = 1000,
  iter = 1000,
  calibrated = TRUE
)
```

Arguments

`case_tasks`	A vector of length 2. The expected case scores from the tasks of interest.
`case_cov`	A vector containing the expected case scores on all covariates included.
`control_tasks`	A 2x2 matrix or dataframe containing the expected means (first column) and standard deviations (second column). Defaults to two variables with means 0 and sd = 1.
`control_covar`	A px2 matrix or dataframe containing the expected means (first column) and standard deviations (second column), p being the number of covariates. Defaults to one covariate with mean 0 and sd = 1.
`cor_mat`	A correlation matrix containing the correlations of the tasks of interest and the coviariate(s). The first two variables are treated as the tasks of interest. Defaults pairwise correlations between the variates of 0.3.
`sample_size`	Single value giving the size of the control sample for which you wish to calculate power.
`alternative`	The alternative hypothesis. A string of either "less", "greater" or "two.sided" (default).
`alpha`	The specified Type I error rate, default is 0.05. This can be varied, with effects on power.
`nsim`	The number of simulations for the power calculation. Defaults to 1000 due to BSDT already being computationally intense. Increase for better accuracy.
`iter`	The number of simulations used by the BSDT_cov, defaults to 1000. Increase for better accuracy.
`calibrated`	Whether or not to use the standard theory (Jeffreys) prior distribution (if set to `FALSE`) or a calibrated prior. See Crawford et al. (2011) for further information. Calibrated prior is recommended.

Value

Returns a single value approximating the power of the test for the given parameters.

Examples

```
BSDT_cov_power(c(-2, 0), case_cov = c(0, 0, 0),
control_covar = matrix(c(0, 0, 0, 1, 1, 1), ncol= 2),
sample_size = 10, cor_mat = diag(5), iter = 20, nsim = 20)
```

`BSDT_power` *Power calculator for BSDT*

Description

Calculates approximate power, given sample size, using Monte Carlo simulation, for specified case scores, means and standard deviations for the control sample. The means and standard deviations default to 0 and 1 respectively, so if no other values are given the case scores are interpreted as deviations from the mean in standard deviations. Hence, the effect size of the dissociation (Z-DCC) would in that case be the difference between the two case scores. Is computationally heavy and might therefore take a few seconds.

Usage

```
BSDT_power(
  case_a,
  case_b,
  mean_a = 0,
  mean_b = 0,
  sd_a = 1,
  sd_b = 1,
  r_ab = 0.5,
  sample_size,
  alternative = c("two.sided", "greater", "less"),
  alpha = 0.05,
  nsim = 1000,
  iter = 1000,
  calibrated = TRUE
)
```

Arguments

`case_a`	A single value from the expected case observation on task A.
`case_b`	A single value from the expected case observation on task B.
`mean_a`	The expected mean from the control sample on task A. Defaults to 0.
`mean_b`	The expected mean from the control sample on task B. Defaults to 0.
`sd_a`	The expected standard deviation from the control sample on task A. Defaults to 1.
`sd_b`	The expected standard deviation from the control sample on task B. Defaults to 1.
`r_ab`	The expected correlation between the tasks. Defaults to 0.5
`sample_size`	The size of the control sample, vary this parameter to see how the sample size affects power.
`alternative`	The alternative hypothesis. A string of either "two.sided" (default) or "one.sided".

`alpha`	The specified Type I error rate. This can be varied, with effects on power. Defaults to 0.05.
`nsim`	The number of simulations to run. Higher number gives better accuracy, but low numbers such as 10000 or even 1000 are usually sufficient for the purposes of this calculator. Defaults to 1000 due to the computationally intense `BSTD`.
`iter`	The number simulations used by `BSTD`. Defaults to 1000.
`calibrated`	Whether or not to use the standard theory (Jeffreys) prior distribution (if set to `FALSE`) or a calibrated prior. See Crawford et al. (2011) for further information. Calibrated prior is recommended.

Value

Returns a single value approximating the power of the test for the given parameters.

Examples

```
BSDT_power(case_a = -3, case_b = -1, mean_a = 0, mean_b = 0,
           sd_a = 1, sd_b = 1, r_ab = 0.5, sample_size = 20, nsim = 100, iter = 100)
```

`BTD` *Bayesian Test of Deficit*

Description

Takes a single observation and compares it to a distribution estimated by a control sample using Bayesian methodology. Calculates standardised difference between the case score and the mean of the controls and proportions falling above or below the case score, as well as associated credible intervals. This approach was developed by Crawford and Garthwaite (2007) but converge to the results of `TD()`, which is faster. Returns the point estimate of the standardised difference between the case score and the mean of the controls and the point estimate of the p-value (i.e. the percentage of the population that would be expected to obtain a lower or higher score, depending on the alternative hypothesis). This test is based on random number generation which means that results may vary between runs. This is by design and the reason for not using `set.seed()` to reproduce results inside the function is to emphasise the randomness of the test. To get more accurate and stable results please increase the number of iterations by increasing `iter` whenever feasible.

Usage

```
BTD(
  case,
  controls,
  sd = NULL,
  sample_size = NULL,
  alternative = c("less", "greater", "two.sided"),
  int_level = 0.95,
  iter = 10000,
  na.rm = FALSE
)
```

Arguments

`case`	Case observation, can only be a single value.
`controls`	Numeric vector of observations from the control sample. If single value, treated as mean.
`sd`	If input of controls is single value, the standard deviation of the sample must be given as well.
`sample_size`	If input of controls is single value, the size of the sample must be given as well.
`alternative`	A character string specifying the alternative hypothesis, must be one of `"less"` (default), `"greater"` or `"two.sided"`. You can specify just the initial letter.
`int_level`	Level of confidence for credible intervals, defaults to 95%.
`iter`	Number of iterations. Set to higher for more accuracy, set to lower for faster calculations.
`na.rm`	Remove `NA`s from controls.

Value

A list with class `"htest"` containing the following components:

`statistic`	the mean z-value over `iter` number of iterations
`parameter`	the degrees of freedom used to specify the posterior distribution.
`p.value`	the mean p-value for all simulated Z-scores.
`estimate`	estimated standardised difference (Z-CC) and point estimate of p-value.
`null.value`	the value of the difference under the null hypothesis.
`interval`	named numerical vector containing credibility level and intervals for both Z-CC and estimated proportion.
`desc`	named numerical containing descriptive statistics: mean and standard deviations of controls as well as sample
`alternative`	a character string describing the alternative hypothesis.
`method`	a character string indicating what type of test was performed.
`data.name`	a character string giving the name(s) of the data as well as summaries.

References

Crawford, J. R., & Garthwaite, P. H. (2007). Comparison of a single case to a control or normative sample in neuropsychology: Development of a Bayesian approach. *Cognitive Neuropsychology, 24*(4), 343-372. doi:10.1080/02643290701290146

Examples

```
BTD(case = -2, controls = 0, sd = 1, sample_size = 20, iter = 1000)

BTD(case = size_weight_illusion[1, "V_SWI"],
    controls = size_weight_illusion[-1, "V_SWI"], alternative = "l", iter = 1000)
```

`BTD_cov` *Bayesian Test of Deficit with Covariates*

Description

Takes a single observation and compares it to a distribution estimated by a control sample, while controlling for the effect of covariates, using Bayesian methodology. This test is used when assessing a case conditioned on some other variable, for example, assessing abnormality when controlling for years of education or sex. Under the null hypothesis the case is an observation from the distribution of scores from the task of interest coming from observations having the same score as the case on the covariate(s). Returns a significance test, point and interval estimates of difference between the case and the mean of the controls as well as point and interval estimates of abnormality, i.e. an estimation of the proportion of controls that would exhibit a more extreme conditioned score. This test is based on random number generation which means that results may vary between runs. This is by design and the reason for not using `set.seed()` to reproduce results inside the function is to emphasise the randomness of the test. To get more accurate and stable results please increase the number of iterations by increasing `iter` whenever feasible. Developed by Crawford, Garthwaite and Ryan (2011).

Usage

```
BTD_cov(
  case_task,
  case_covar,
  control_task,
  control_covar,
  alternative = c("less", "two.sided", "greater"),
  int_level = 0.95,
  iter = 10000,
  use_sumstats = FALSE,
  cor_mat = NULL,
  sample_size = NULL
)
```

Arguments

`case_task`	The case score from the task of interest. Must be a single value.
`case_covar`	A vector containing the case scores on all covariates included. Can be of any length except 0, in that case use `BTD`.

`control_task`	A vector containing the scores from the controls on the task of interest. Or a vector of length 2 containing the mean and standard deviation of the task. In that order.
`control_covar`	A vector, matrix or dataframe containing the control scores on the covariates included. If matrix or dataframe each column represents a covariate. Or a matrix or dataframe containing summary statistics where the first column represents the means for each covariate and the second column represents the standard deviation.
`alternative`	A character string specifying the alternative hypothesis, must be one of `"two.sided"` (default), `"greater"` or `"less"`. You can specify just the initial letter.
`int_level`	The probability level on the Bayesian credible intervals, defaults to 95%.
`iter`	Number of iterations to be performed. Greater number gives better estimation but takes longer to calculate. Defaults to 10000.
`use_sumstats`	If set to `TRUE`, `control_tasks` and `control_covar` are treated as matrices with summary statistics. Where the first column represents the means for each variable and the second column represents the standard deviation.
`cor_mat`	A correlation matrix of all variables included. NOTE: the first variable should be the task of interest.
`sample_size`	An integer specifying the sample size of the controls.

Details

Uses random generation of inverse wishart distributions from the CholWishart package (Geoffrey Thompson, 2019).

Value

A list with class `"htest"` containing the following components:

`statistic`	the average z-value over `iter` number of iterations.
`parameter`	the degrees of freedom used to specify the posterior distribution.
`p.value`	the average p-value over `iter` number of iterations.
`estimate`	case scores expressed as z-scores on task X and Y. Standardised effect size (Z-CCC) of task difference betwee
`null.value`	the value of the difference between tasks under the null hypothesis.
`interval`	named numerical vector containing level of confidence and confidence intervals for both effect size and p-valu
`desc`	data frame containing means and standard deviations for controls as well as case scores.
`cor.mat`	matrix giving the correlations between the task of interest and the covariates included.
`sample.size`	number of controls..

`alternative`	a character string describing the alternative hypothesis.
`method`	a character string indicating what type of test was performed.
`data.name`	a character string giving the name(s) of the data

References

Crawford, J. R., Garthwaite, P. H., & Ryan, K. (2011). Comparing a single case to a control sample: Testing for neuropsychological deficits and dissociations in the presence of covariates. *Cortex, 47*(10), 1166-1178. doi:10.1016/j.cortex.2011.02.017

Geoffrey Thompson (2019). CholWishart: Cholesky Decomposition of the Wishart Distribution. R package version 1.1.0. `https://CRAN.R-project.org/package=CholWishart`

Examples

```
BTD_cov(case_task = size_weight_illusion[1, "V_SWI"],
        case_covar = size_weight_illusion[1, "YRS"],
        control_task = size_weight_illusion[-1, "V_SWI"],
        control_covar = size_weight_illusion[-1, "YRS"], iter = 100)
```

`BTD_cov_power` *Power calculator for BTD_cov*

Description

Computationally intense. Lower `iter` and/or `nsim` for less exact but faster calculations. Calculates approximate power, given sample size, using Monte Carlo simulation for the Bayesian test of deficit with covariates for specified (expected) case score, means and standard deviations for the control sample on the task of interest and included covariates. The number of covariates defaults to 1, means and standard deviations for the task and covariate defaults to 0 and 1, so if no other values are given the case score is interpreted as deviation from the mean in standard deviations for both task and covariate.

Usage

```
BTD_cov_power(
  case,
  case_cov,
  control_task = c(0, 1),
  control_covar = c(0, 1),
  cor_mat = diag(2) + 0.3 - diag(c(0.3, 0.3)),
  sample_size,
  alternative = c("less", "greater", "two.sided"),
  alpha = 0.05,
  nsim = 1000,
  iter = 1000
)
```

Arguments

`case`	A single value from the expected case observation on the task of interest.
`case_cov`	A vector of expected case observations from covariates of interest.
`control_task`	A vector of length 2 containing the expected mean and standard deviation of the task of interest. In that order.
`control_covar`	A matrix with 2 columns containing expected means (in the 1st column) and standard deviations (in the 2nd column) of the included covariates.
`cor_mat`	A correlation matrix containing the correlations of the task of interest and the coviariate(s). The first variable is treated as the task of interest. Defaults to a correlation of 0.3 between the covariate and the variate of interest.
`sample_size`	Single value of the size of the sample for which you wish to calculate power.
`alternative`	The alternative hypothesis. A string of either "less" (default), "greater" or "two.sided".
`alpha`	The specified Type I error rate. This can also be varied, with effects on power.
`nsim`	The number of simulations for the power calculation. Defaults to 1000 due to BTD_cov already being computationally intense.
`iter`	The number of simulations used by the BTD_cov. Defaults to 1000.

Value

Returns a single value approximating the power of the test for the given parameters.

Examples

```
cor_mat = matrix(c(1, 0.2, 0.3, 0.2, 1, 0.4, 0.3, 0.4, 1), ncol = 3)

BTD_cov_power(case = -2, case_cov = c(105, 30), control_task = c(0, 1),
control_covar = matrix(c(100, 40, 15, 10), ncol = 2), sample_size = 15,
cor_mat = cor_mat, iter = 20, nsim = 20)
```

`BTD_power` *Power calculator for BTD*

Description

Calculates approximate power, given sample size, using Monte Carlo simulation for the Bayesian test of deficit for a specified case score, mean and standard deviation for the control sample. The mean and standard deviation defaults to 0 and 1, so if no other values are given the case score is interpreted as deviation from the mean in standard deviations.

Usage

```
BTD_power(
  case,
  mean = 0,
  sd = 1,
  sample_size,
  alternative = c("less", "greater", "two.sided"),
  alpha = 0.05,
  nsim = 1000,
  iter = 1000
)
```

Arguments

`case`	A single value from the expected case observation.
`mean`	The expected mean of the control sample.
`sd`	The expected standard deviation of the control sample.
`sample_size`	The size of the control sample, vary this parameter to see how the sample size affects power.
`alternative`	The alternative hypothesis. A string of either "less" (default), "greater" or "two.sided".
`alpha`	The specified Type I error rate. This can also be varied, with effects on power.
`nsim`	The number of simulations for the power calculation. Defaults to 1000 due to BTD already being computationally intense.
`iter`	The number of simulations used by the BTD. Defaults to 1000.

Value

Returns a single value approximating the power of the test for the given parameters.

Examples

```
BTD_power(case = -2, mean = 0, sd = 1, sample_size = 20)
```

`MTD` *Multivariate Test of deficit*

Description

Testing for abnormality in the distance between a a vector of observations for a single case and a vector of population means. Please see vignette for further details.

Usage

```
MTD(
  case,
  controls,
  conf_level = 0.95,
  method = c("pd", "pchi", "pf", "pmd"),
  mahalanobis_dist = NULL,
  k = NULL,
  n = NULL
)
```

Arguments

`case`	Vector of case scores
`controls`	Matrix or data frame with scores from the control sample, each column representing a variable
`conf_level`	Level of confidence for the confidence intervals
`method`	One out of "pd", "pchi", "pf" and "pmd". Use "pmd" if the Mahalanobi's distance seems suspiciously small
`mahalanobis_dist`	Mahalanobi's distance of the case if summary statistics are used
`k`	The number of dimensions, if summary statistics are used
`n`	The size of the control sample

Value

A list with class `"htest"` containing the following components:

`statistic`	Hotelling's T^2 statistic for the case's Mahalanobi's distance
`p.value`	The p value associated with the Hotelling statistic
`estimate`	Estimates of the case Mahalanobis distance and index as well as abnormality
`interval`	List of interval measure for the estimates
`sample.size`	number of controls.
`method`	a character string indicating what type of test was performed and which abnormality measure used

Examples

```
caseA <- size_weight_illusion[1, "V_SWI"]
contA <- size_weight_illusion[-1, "V_SWI"]
caseB <- size_weight_illusion[1, "K_SWI"]
contB <- size_weight_illusion[-1, "K_SWI"]
```

```
MTD(case = c(caseA, caseB), controls = cbind(contA, contB),
  conf_level = 0.95, method = c("pd", "pchi", "pf", "pmd"),
  mahalanobis_dist = NULL, k = NULL, n = NULL)
```

RSDT *Revised Standardised Difference Test*

Description

A test on the discrepancy between two tasks in a single case, by comparison to the discrepancy of means in the same two tasks in a control sample. Standardises task scores as well as task discrepancy, so the tasks do not need to be measured on the same scale. Calculates a standardised effect size (Z-DCC) of task discrepancy as well as a point estimate of the proportion of the control population that would be expected to show a more extreme discrepancy. Developed by Crawford and Garthwaite (2005).

Usage

```
RSDT(
  case_a,
  case_b,
  controls_a,
  controls_b,
  sd_a = NULL,
  sd_b = NULL,
  sample_size = NULL,
  r_ab = NULL,
  alternative = c("two.sided", "greater", "less"),
  na.rm = FALSE
)
```

Arguments

`case_a`	Case's score on task A.
`case_b`	Case's score on task B.
`controls_a`	Controls' scores on task A. Takes either a vector of observations or a single value interpreted as mean. *Note*: you can supply a vector as input for task A while mean and SD for task B.
`controls_b`	Controls' scores on task B. Takes either a vector of observations or a single value interpreted as mean. *Note*: you can supply a vector as input for task B while mean and SD for task A.
`sd_a`	If single value for task A is given as input you must supply the standard deviation of the sample.

`sd_b`	If single value for task B is given as input you must supply the standard deviation of the sample.
`sample_size`	If A or B is given as mean and SD you must supply the sample size. If controls_a is given as vector and controls_b as mean and SD, sample_size must equal the number of observations in controls_a.
`r_ab`	If A or B is given as mean and SD you must supply the correlation between the tasks.
`alternative`	A character string specifying the alternative hypothesis, must be one of `"two.sided"` (default), `"greater"` or `"less"`. You can specify just the initial letter. Since the direction of the expected effect depends on which task is set as A and which is set as B, be very careful if changing this parameter.
`na.rm`	Remove `NA`s from controls.

Value

A list with class `"htest"` containing the following components:

`statistic`	Returns the value of a approximate t-statistic, however, because of the underlying equation, it cannot be negat
`parameter`	the degrees of freedom for the t-statistic.
`p.value`	the p-value for the test.
`estimate`	case scores expressed as z-scores on task A and Y. Standardised effect size (Z-DCC) of task difference betwee
`sample.size`	the size of the control sample
`null.value`	the value of the discrepancy under the null hypothesis.
`alternative`	a character string describing the alternative hypothesis.
`method`	a character string indicating what type of test was performed.
`data.name`	a character string giving the name(s) of the data

References

Crawford, J. R., & Garthwaite, P. H. (2005). Testing for Suspected Impairments and Dissociations in Single-Case Studies in Neuropsychology: Evaluation of Alternatives Using Monte Carlo Simulations and Revised Tests for Dissociations. *Neuropsychology, 19*(3), 318 - 331. doi:10.1037/08944105.19.3.318

Examples

```
RSDT(-3.857, -1.875, controls_a = 0, controls_b = 0, sd_a = 1,
sd_b = 1, sample_size = 20, r_ab = 0.68)

RSDT(case_a = size_weight_illusion[1, "V_SWI"], case_b = size_weight_illusion[1, "K_SWI"],
```

```
controls_a = size_weight_illusion[-1, "V_SWI"], controls_b = size_weight_illusion[-1, "K_SWI"])
```

`RSDT_power` *Power calculator for RSDT*

Description

Calculates approximate power, given sample size, using Monte Carlo simulation, for specified case scores, means and standard deviations for the control sample. The means and standard deviations defaults to 0 and 1 respectively, so if no other values are given the case scores are interpreted as deviations from the mean in standard deviations. Hence, the effect size of the dissociation (Z-DCC) would in that case be the difference between the two case scores.

Usage

```
RSDT_power(
  case_a,
  case_b,
  mean_a = 0,
  mean_b = 0,
  sd_a = 1,
  sd_b = 1,
  r_ab = 0.5,
  sample_size,
  alternative = c("two.sided", "greater", "less"),
  alpha = 0.05,
  nsim = 10000
)
```

Arguments

`case_a`	A single value from the expected case observation on task A.
`case_b`	A single value from the expected case observation on task B.
`mean_a`	The expected mean from the control sample on task A. Defaults to 0.
`mean_b`	The expected mean from the control sample on task B. Defaults to 0.
`sd_a`	The expected standard deviation from the control sample on task A. Defaults to 1.
`sd_b`	The expected standard deviation from the control sample on task B. Defaults to 1.
`r_ab`	The expected correlation between the tasks. Defaults to 0.5
`sample_size`	The size of the control sample, vary this parameter to see how the sample size affects power.
`alternative`	The alternative hypothesis. A string of either "two.sided" (default) or "one.sided".

`alpha`	The specified Type I error rate. This can also be varied, with effects on power. Defaults to 0.05.
`nsim`	The number of simulations to run. Higher number gives better accuracy, but low numbers such as 10000 or even 1000 are usually sufficient for the purposes of this calculator.

Value

Returns a single value approximating the power of the test for the given parameters.

Examples

```
RSDT_power(case_a = -3, case_b = -1, mean_a = 0, mean_b = 0,
           sd_a = 1, sd_b = 1, r_ab = 0.5, sample_size = 20, nsim = 1000)
```

`singcar` *singcar: Comparing Single Cases to Small Samples*

Description

The aim of **singcar** is to provide and encourage usage of appropriate statistical methods for comparing a case against a control sample. For instance, they may commonly be done in a neuropsychological context, in which an individual has incurred a specific brain injury and we wish to test whether this damage has led to an impairment of some cognitive function and whether two different functions are dissociable. For many cognitive functions there is normed data available which the patient can be compared against directly. However, when this is not possible a control sample estimating the population, against which we wish to compare the patient, must be used. Both frequentist and Bayesian methods have been developed to do this, first and foremost by John Crawford and Paul Garthwaite (Crawford et al., 2011; Crawford & Garthwaite, 2002, 2007, 2005; Crawford & Howell, 1998). It is these methods that **singcar** implements. Power calculators for each respective test are also provided. Although the canonical applications for these tests are in Cognitive Neuropsychology or Clinical Neuropsychology, they are potentially applicable to any circumstance in which a measure taken from a single individual is to be compared against data from a normative sample (i.e. a control group). It should be noted that these statistical methods could also be applied as a general method of outlier detection in small samples.

singcar functions

`TD()`

`BTD()`

`BTD_cov()`

`RSDT()`

`UDT()`

`BSDT()`

`BSDT_cov()`

`TD_power()`

`BTD_power()`

`BTD_cov_power()`

`RSDT_power()`

`UDT_power()`

`BSDT_power()`

`BSDT_cov_power()`

References

Crawford, J., & Garthwaite, P. (2002). Investigation of the single case in neuropsychology: Confidence limits on the abnormality of test scores and test score differences. Neuropsychologia, 40(8), 1196-1208. https://doi.org/10.1016/S0028-3932(01)00224-X

Crawford, J., & Garthwaite, P. (2007). Comparison of a single case to a control or normative sample in neuropsychology: Development of a Bayesian approach. Cognitive Neuropsychology, 24(4), 343-372. https://doi.org/10.1080/02643290701290146

Crawford, J., & Garthwaite, P. (2005). Testing for Suspected Impairments and Dissociations in Single-Case Studies in Neuropsychology: Evaluation of Alternatives Using Monte Carlo Simulations and Revised Tests for Dissociations. Neuropsychology, 19(3), 318-331. https://doi.org/10.1037/0894-4105.19.3.318

Crawford, J., Garthwaite, P., & Ryan, K. (2011). Comparing a single case to a control sample: Testing for neuropsychological deficits and dissociations in the presence of covariates. Cortex, 47(10), 1166-1178. https://doi.org/10.1016/j.cortex.2011.02.017

Crawford, J., & Howell, D. (1998). Comparing an Individual's Test Score Against Norms Derived from Small Samples. The Clinical Neuropsychologist, 12(4), 482-486. https://doi.org/10.1076/clin.12.4.482.7241

`size_weight_illusion` *Data from one patient and 28 controls on the size-weight illusion*

Description

A dataset containing data from 28 healthy controls and one patient, DF, with visual form agnosia (inability to perceive the form of objects) from bilateral lesions to the lateral occipital complex. The size-weight illusion occurs when a person underestimates the weight of a larger item when compared to a smaller of equal weight (Charpentier, 1891). From these data, one can assess the magnitude of the illusion for patient DF by comparison to age-matched controls under visual and kinaesthetic cue conditions. The measure of the size-weight illusion is a scaled measure expressing the number of grams weight difference perceived per cubic cm of volume change (Hassan et al, 2020).

Usage

```
size_weight_illusion
```

Format

A data frame with 29 rows and 6 variables:

GROUP factor with patient (SC) or control group (HC)

PPT participant identifier

SEX gender of partcipants

YRS age of participants

V_SWI SWI measure from the visual task

K_SWI SWI measure from the kinaesthetic task

Source

`https://osf.io/3s2fp/?view_only=50c8af0b39ee436b85d292b0a701cc3b`

References

Hassan, E. K., Sedda, A., Buckingham, G., & McIntosh, R. D. (2020). The size-weight illusion in visual form agnosic patient DF. Neurocase, 1-8. https://doi.org/10.1080/13554794.2020.1800748

`TD` *Test of Deficit*

Description

Crawford and Howell's (1998) modified t-test. Takes a single observation and compares it to a distribution estimated by a control sample. Calculates standardised difference between the case score and the mean of the controls and proportions falling above or below the case score, as well as associated confidence intervals.

Usage

```
TD(
  case,
  controls,
  sd = NULL,
  sample_size = NULL,
  alternative = c("less", "greater", "two.sided"),
  conf_int = TRUE,
  conf_level = 0.95,
  conf_int_spec = 0.01,
  na.rm = FALSE
)
```

Arguments

`case`	Case observation, can only be a single value.
`controls`	Numeric vector of observations from the control sample. If single value, treated as mean.
`sd`	If input of controls is single value, the standard deviation of the sample must be given as well.
`sample_size`	If input of controls is single value, the size of the sample must be gven as well.
`alternative`	A character string specifying the alternative hypothesis, must be one of `"less"` (default), `"greater"` or `"two.sided"`. You can specify just the initial letter.
`conf_int`	Initiates a search algorithm for finding confidence intervals. Defaults to `TRUE`, set to `FALSE` for faster calculation (e.g. for simulations).
`conf_level`	Level of confidence for intervals, defaults to 95%.
`conf_int_spec`	The size of iterative steps for calculating confidence intervals. Smaller values gives more precise intervals but takes longer to calculate. Defaults to a specificity of 0.01.
`na.rm`	Remove `NA`s from controls.

Details

Returns the point estimate of the standardised difference between the case score and the mean of the controls and the point estimate of the p-value (i.e. the percentage of the population that would be expected to obtain a lower or higher score, depending on the alternative hypothesis).

Value

A list of class `"htest"` containing the following components:

`statistic`	the value of the t-statistic.
`parameter`	the degrees of freedom for the t-statistic.
`p.value`	the p-value for the test.
`estimate`	estimated standardised difference (Z-CC) and point estimate of p-value.
`null.value`	the value of the difference under the null hypothesis.
`interval`	named numerical vector containing level of confidence and confidence intervals for both Z-CC and p-value.
`desc`	named numerical containing descriptive statistics: mean and standard deviations of controls as well as sample
`alternative`	a character string describing the alternative hypothesis.
`method`	a character string indicating what type of t-test was performed.
`data.name`	a character string giving the name(s) of the data as well as summaries.

Note of caution

Calculating the confidence intervals relies on finding non-centrality parameters for non-central t-distributions. Depending on the degrees of freedom, the confidence level and the effect size exact accuracy from the `stats::qt()` function used can not be guaranteed. However, the approximations should be good enough for most cases. See `https://stat.ethz.ch/pipermail/r-help/2008-June/164843.html`.

References

Crawford, J. R., & Howell, D. C. (1998). Comparing an Individual's Test Score Against Norms Derived from Small Samples. *The Clinical Neuropsychologist, 12*(4), 482 - 486. doi:10.1076/clin.12.4.482.7241

Crawford, J. R., & Garthwaite, P. H. (2002). Investigation of the single case in neuropsychology: Confidence limits on the abnormality of test scores and test score differences. *Neuropsychologia, 40*(8), 1196-1208. doi:10.1016/S00283932(01)00224X

Examples

```
TD(case = -2, controls = 0, sd = 1, sample_size = 20)

TD(case = size_weight_illusion[1, "V_SWI"],
   controls = size_weight_illusion[-1, "V_SWI"], alternative = "l")
```

`TD_power` *Power calculator for TD*

Description

Calculates exact power given sample size or sample size given power, using analytical methods for the frequentist test of deficit for a specified case score and mean and standard deviation for the control sample. The mean and standard deviation defaults to 0 and 1, so if no other values are given the case score is interpreted as deviation from the mean in standard deviations.

Usage

```
TD_power(
  case,
  mean = 0,
  sd = 1,
  sample_size = NULL,
  power = NULL,
  alternative = c("less", "greater", "two.sided"),
  alpha = 0.05,
  spec = 0.005
)
```

Arguments

`case`	A single value from the expected case observation.
`mean`	The expected mean of the control sample.
`sd`	The expected standard deviation of the control sample.
`sample_size`	The size of the control sample, vary this parameter to see how the sample size affects power. One of sample size or power must be specified, not both.
`power`	A single value between 0 and 1 specifying desired power for calculating necessary sample size. One of sample size or power must be specified, not both.
`alternative`	The alternative hypothesis. A string of either "less" (default), "greater" or "two.sided".
`alpha`	The specified Type I error rate. This can also be varied, with effects on power.
`spec`	A single value between 0 and 1. If desired power is given as input the function will utilise a search algorithm to find the sample size needed to reach the desired power. However, if the power specified is greater than what is actually possible to achieve the algorithm could search forever. Hence, when power does not increase substantially for any additional participant in the sample, the algorithm stops. By default the algorithm stops when power does not increase more than 0.5% for any added participant, but by varying `spec`, this specificity can be changed.

Value

Either a single value of the exact power, if sample size is given. Or a dataframe consisting of both the sample size and the exact power such size would yield.

Examples

```
TD_power(case = -2, mean = 0, sd = 1, sample_size = 20)
TD_power(case = -2, mean = 0, sd = 1, power = 0.8)
```

`UDT` *Unstandardised Difference Test*

Description

A test on the discrepancy between two tasks in a single case, by comparison to the mean of discrepancies of the same two tasks in a control sample. Use *only* when the two tasks are measured on the same scale with the same underlying distribution because no standardisation is performed on task scores. As a rule-of-thumb, the UDT may be applicable to pairs of tasks for which it would be sensible to perform a paired t-test within the control group. Calculates however a standardised effect size in the same manner as `RSDT()`. This is original behaviour from Crawford and Garthwaite (2005) but might not be appropriate. So use this standardised effect size with caution. Calculates a standardised effect size of task discrepancy as well as a point estimate of the proportion of the control population that would be expected to show a more extreme discrepancy and respective confidence intervals.

Usage

```
UDT(
  case_a,
  case_b,
  controls_a,
  controls_b,
  sd_a = NULL,
  sd_b = NULL,
  sample_size = NULL,
  r_ab = NULL,
  alternative = c("two.sided", "greater", "less"),
  conf_int = TRUE,
  conf_level = 0.95,
  conf_int_spec = 0.01,
  na.rm = FALSE
)
```

Arguments

`case_a`	Case's score on task A.
`case_b`	Case's score on task B.
`controls_a`	Controls' scores on task A. Takes either a vector of observations or a single value interpreted as mean. *Note*: you can supply a vector as input for task A while mean and SD for task B.
`controls_b`	Controls' scores on task B. Takes either a vector of observations or a single value interpreted as mean. *Note*: you can supply a vector as input for task B while mean and SD for task A.
`sd_a`	If single value for task A is given as input you must supply the standard deviation of the sample.
`sd_b`	If single value for task B is given as input you must supply the standard deviation of the sample.
`sample_size`	If A or B is given as mean and SD you must supply the sample size. If controls_a is given as vector and controls_b as mean and SD, sample_size must equal the number of observations in controls_a.
`r_ab`	If A and/or B is given as mean and SD you must supply the correlation between the tasks.
`alternative`	A character string specifying the alternative hypothesis, must be one of `"two.sided"` (default), `"greater"` or `"less"`. You can specify just the initial letter. Since the direction of the expected effect depends on which task is set as A and which is set as B, be very careful if changing this parameter.
`conf_int`	Initiates a search algorithm for finding confidence intervals. Defaults to `TRUE`, set to `FALSE` for faster calculation (e.g. for simulations).
`conf_level`	Level of confidence for intervals, defaults to 95%.
`conf_int_spec`	The size of iterative steps for calculating confidence intervals. Smaller values gives more precise intervals but takes longer to calculate. Defaults to a specificity of 0.01.

`na.rm`	Remove `NA`s from controls.

Details

Running `UDT` is equivalent to running `TD` on discrepancy scores making it possible to run unstandardised tests with covariates by applying `BTD_cov` to discrepancy scores.

Value

A list with class `"htest"` containing the following components:

`statistic`	the t-statistic.
`parameter`	the degrees of freedom for the t-statistic.
`p.value`	the p-value of the test.
`estimate`	unstandardised case scores, task difference and pont estimate of proportion control population expected to ab
`control.desc`	named numerical with descriptive statistics of the control samples.
`null.value`	the value of the difference under the null hypothesis.
`alternative`	a character string describing the alternative hypothesis.
`method`	a character string indicating what type of test was performed.
`data.name`	a character string giving the name(s) of the data

References

Crawford, J. R., & Garthwaite, P. H. (2005). Testing for Suspected Impairments and Dissociations in Single-Case Studies in Neuropsychology: Evaluation of Alternatives Using Monte Carlo Simulations and Revised Tests for Dissociations. *Neuropsychology, 19*(3), 318 - 331. doi:10.1037/08944105.19.3.318

Examples

```
UDT(-3.857, -1.875, controls_a = 0, controls_b = 0, sd_a = 1,
sd_b = 1, sample_size = 20, r_ab = 0.68)

UDT(case_a = size_weight_illusion[1, "V_SWI"], case_b = size_weight_illusion[1, "K_SWI"],
 controls_a = size_weight_illusion[-1, "V_SWI"], controls_b = size_weight_illusion[-1, "K_SWI"])
```

`UDT_power` *Power calculator for UDT*

Description

Calculates exact power given sample size or sample size given power, using analytical methods for the frequentist test of deficit for a specified case scores, means and standard deviations for the control sample. The means and standard deviations defaults to 0 and 1 respectively, so if no other values are given, the case scores are interpreted as deviations from the mean in standard deviations. The returned value will approximate the power for the given parameters.

Usage

```
UDT_power(
  case_a,
  case_b,
  mean_a = 0,
  mean_b = 0,
  sd_a = 1,
  sd_b = 1,
  r_ab = 0.5,
  sample_size = NULL,
  power = NULL,
  alternative = c("two.sided", "greater", "less"),
  alpha = 0.05,
  spec = 0.005
)
```

Arguments

`case_a`	A single value from the expected case observation on task A.
`case_b`	A single value from the expected case observation on task B.
`mean_a`	The expected mean from the control sample on task A. Defaults to 0.
`mean_b`	The expected mean from the control sample on task B. Defaults to 0.
`sd_a`	The expected standard deviation from the control sample on task A. Defaults to 1.
`sd_b`	The expected standard deviation from the control sample on task B. Defaults to 1.
`r_ab`	The expected correlation between the tasks. Defaults to 0.5
`sample_size`	The size of the control sample, vary this parameter to see how the sample size affects power. One of sample size or power must be specified, not both.
`power`	A single value between 0 and 1 specifying desired power for calculating necessary sample size. One of sample size or power must be specified, not both.
`alternative`	The alternative hypothesis. A string of either "two.sided" (default) or "one.sided".

`alpha`	The specified Type I error rate. This can also be varied, with effects on power. Defaults to 0.05.
`spec`	A single value between 0 and 1. If desired power is given as input the function will utilise a search algorithm to find the sample size needed to reach the desired power. However, if the power specified is greater than what is actually possible to achieve the algorithm could search forever. Hence, when power does not increase substantially for any additional participant in the sample, the algorithm stops. By default the algorithm stops when power does not increase more than 0.5 participant, but by varying `spec`, this specificity can be changed.

Value

Either a single value of the exact power, if sample size is given. Or a dataframe consisting of both the sample size and the exact power such size would yield.

Examples

```
UDT_power(case_a = -3, case_b = -1, mean_a = 0, mean_b = 0,
          sd_a = 1, sd_b = 1, r_ab = 0.5, sample_size = 20)
UDT_power(case_a = -3, case_b = -1, power = 0.8)
```

Index